⑲日本国特許庁（JP）　⑪特許出願公開

⑫公開特許公報（A）　平1−276189

⑤1Int.Cl.⁴	識別記号	庁内整理番号	⑬公開　平成1年(1989)11月6日
G 09 C 1/00		7368−5B	

審査請求　未請求　請求項の数 8　（全9頁）

⑤4発明の名称　暗号方式

⑳特　願　昭63−103919
㉒出　願　昭63(1988)4月28日

⑦2発明者　宝木　和夫　神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
⑦2発明者　中川　聡夫　茨城県日立市大みか町5丁目2番1号　株式会社日立コントロールシステムズ内
⑦2発明者　佐々木　良一　神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
⑦1出願人　株式会社日立製作所　東京都千代田区神田駿河台4丁目6番地
⑦1出願人　株式会社日立コントロールシステムズ　茨城県日立市大みか町5丁目2番1号
⑦4代理人　弁理士　小川　勝男　外1名

明　細　書

1．発明の名称

暗号方式

2．特許請求の範囲

1．換字処理とデータ撹乱のための転置処理を組合せて一定長のデータを暗号化するブロック暗号方式において、上記転置処理は、2^nビット（n＝2以上の整数）だけ右または左に循環シフトするという操作を組合せることにより行なうことを特徴とする暗号方式。

2．上記換字処理は、ある定数をXとして、Xビットのデータと Xビットの鍵データとの間で演算を行なうという操作と、Xビットのデータと該データを右または左に2ビット循環シフトしたものと定数Xとの和を2^Xで割った余りをとるという操作を、組合せることにより行なうことを特徴とする第1項の暗号方式。

3．上記換字処理と転置処理を組み合わせた暗号処理は、暗号化されたデータと、次の暗号化されるべきデータとの間で演算処理を施した後、該演算結果をさらに暗号化するというフィードバック処理を有することを特徴とする第1項または第2項の暗号方式。

4．上記転置処理は、32ビットマイクロコンピュータのソフトウェアを用いて、32ビットのデータを、4ビット、または8ビット、または16ビットだけ右または左に循環シフトするという操作を組合せて行なうことを特徴とする第1項〜第3項いずれか1項の暗号方式。

5．上記転置処理は、16ビットマイクロコンピュータのソフトウェアを用いて、16ビットのデータを、4ビット、または8ビット、右または左に循環するという操作を組合せて行なうことを特徴とする第1項〜第3項いづれか1項の暗号方式。

6．上記転置処理は、8ビットマイクロコンピュータのソフトウェアを用いて、8ビットのデータを、4ビット右または左に循環シフトするという操作により行なうことを特徴とする第1項〜第3項いづれか1項の暗号方式。

7．上記換字処理と転置処理とを組み合わせて、メッセージ認証機能を実現するディジタル署名におけるメッセージ認証コード生成のために必要となるメッセージの圧縮文を生成することを特徴とする第1項～第3項いづれか1項の暗号方式。

8．換字処理とデータ撹乱のための転置処理を組合せて一定長のデータを暗号化するブロック暗号方式において、上記転置処理は、2^nビット（n＝2以上の整数）だけ右または左に循環シフトするという操作を組合せることにより行ない、上記換字処理は、ある定数をXとして、Xビットのデータと Xビットの鍵データとの間で演算を行なうという操作と、Xビットのデータと該データを右または左に2ビット循環シフトしたものと定数Xとの和を2^Xで割った余りをとるという操作を、組合せることにより行ない、メッセージ認証コードを生成するための圧縮暗号に、上記換字処理と転置処理のすくなくとも一つを用いることにより、メッセージ認証コードの生成と確認をおこなう機能をＩＣカードに組み入れたことを特徴とする暗号方式。

3．発明の詳細な説明

〔産業上の利用分野〕

本発明は、コンピュータのメッセージ等を暗号化する装置に関する。

〔従来の技術〕

従来の代表的な暗号アルゴリズムとしては、ＤＥＳ（Data Encryption Standard）とＦＥＡＬ（Fast Encryption Standard）が知られており、ＤＥＳに関しては例えば、（1）小山他、「現代暗号理論」、電子通信学会、ｐｐ．４１～４９．昭和６１年９月において、また、ＦＥＡＬに関しては、（2）清水他、「高速データ暗号アルゴリズムＦＥＡＬ」、電子通信学会論文誌Ｄ、Ｖｏｌ．Ｊ７０－Ｄ、№７、ｐｐ．１４１３～１４２３、１９８７年７月において、それぞれ詳細に述べられている。

先ず、ＤＥＳの処理における非線形の計算部分、つまりＳボックスといわれる処理について説明する（上記（1）のｐ．４５、図３－２とｐ４６、図３－３参照）。３２ビットのRは、まず、表1に示す拡大型転置表によって置き換えられると共に、一部のビットは重複されて48ビットに拡大されている。

表1　Sボックスの換字表（S_1）

行＼列	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	9	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

このようにして得られた４８ビットのRは、頭から4ビットごとにその後の2ビットを加えた次のような6ビットずつの8組のブロックを形成している。

r_{31}　r_1　r_2　r_3　r_4　r_5，

r_4　r_5　r_6　r_7　r_8　r_9，

r_8　r_9　r_{10}　r_{11}　r_{12}　r_{13}，

r_{12}　r_{13}　……，……，……　r_{28}　r_{29}，

r_{28}　r_{29}　r_{30}　r_{31}　r_{32}　r_1，

この４８ビットのR'は、同じく４８ビットの鍵Kと排他的論理和の演算を行ない、6ビットずつ8組に分割して、S_1からS_8までの8つのSボックスに入力する。S_1～S_8を選択関数と呼ぶ。これらのSボックスは、6ビットを入力して4ビットを出力する。

例として、表2に一つのSボックスS_1を取り上げてその換字表を示す。

一つのSボックスには、4種類（行番号0，1，2，3）が用意され、この4種類の換字表のどれを用いるかは、入力した6ビットのうち最初と最後のビットを用いて換字表を選ぶ。そして選ばれた換字表にしたがって入力した6ビットの中央の4ビットが換字される。具体的な例として、S_1に対して2進数の入力パターンが０１１０１１となっている場合、最初の０と最後の１で表わされ

表2　拡大型転置表　E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

ている０１、つまり行１（２進数０１は１０進級１であるから）の換字表が選ばれる。次に中央の４ビットのパターン１１０１（１０進数１３）で表わされる列１３で指定され、この結果行１、列１３で指定される値５、つまり０１０１が出力されて４ビットの換字パターンとなる。ＤＥＳではこのような処理ｆ（Ｒ，Ｋ）を用い一段の処理を構成し、これを１６段繰り返す。

上記の処理例に見られるように、ＤＥＳは１ビット単位の処理が基本になっている。

次にＦＥＡＬの処理における非線形の計算部分、つまり、関数Ｓを含んでいる部分について説明する（上記（２）のｐ．１４１６，図４及び図５参照）．ＦＥＡＬの非線形部はＤＥＳの非線形部に比べ、数学的な記述が簡単である。３２ビットデータαは８ビットのデータα^0，α^1，α^2，α^3にそれぞれ分割された後、８ビットを単位として、鍵データと排他的論理和がとられる。その後、所定の関数Ｓによる処理が実行される。

関数Ｓ：$S(x_1 + x_2 + \delta) = Rot_2(w)$

ただし、$w = (x_1 + x_2 + \delta) \bmod 256$

$\delta = 0$または１（定数）

この処理ｆ（α，β）を用い、一段の処理を構成し、これが８段繰り返される。上記の処理に見られるように、ＦＥＡＬは８ビット単位の処理が基本になっている。

〔発明が解決しようとする問題点〕

情報処理と通信技術の進歩によるコンピュータ・ネットワークの普及化、大衆化に伴い、データの不正使用や盗取等に対する情報セキュリティを確保するため、伝送路上のデータやコンピュータに蓄積されたデータを暗号化することは有効な対策であると考えられる。

昭和５２年に、米国商務省標準局が暗号アルゴリズムの標準として制定したＤＥＳは、データの暗号化を行う一つの手段である。

ところが、ＤＥＳはビット単位での処理がたいへん多いため、バイト単位の処理を基調とするマイクロコンピュータのソフトで実現しようとすると、処理に時間がかかり、実用的な速度が得られなかった。

この問題に対し、上記ＦＥＡＬは、１バイト（８ビット）単位の処理を基調とするため、８ビットマイクロコンピュータで実現する場合、ＤＥＳに比べ数倍以上の高速化を達成することができた。ＦＥＡＬにより、８ビットマイクロコンピュータのソフトを用いてある程度実用的な速度が得られるようになったと考えられる。

しかし、最近のマイクロエレクトロニクスの技術の進歩によって、８ビットマイクロコンピュータよりも１６ビットマイクロコンピュータ、さらに、１６ビットマイクロコンピュータよりも３２ビットマイクロコンピュータが使われ出している。近い将来、３２ビットマイクロコンピュータが使われる割合がたいへん大きくなると予想されている。３２ビットマイクロコンピュータの時代になると、さらに高速の暗号処理が要求されるものと予想される。ところが、３２ビットマイクロコンピュータは４バイト基調の処理を行うため、１バイト基調の８ビットマイクロコンピュータ用に設計されたＦＥＡＬを３２ビットマイクロコンピュータで実現しようとすると非効率であった。

そこで、３２ビットマイクロコンピュータ向けの４バイト基調の処理を行う暗号アルゴリズムが望まれていた。

〔問題点を解決するための手段〕

上記の問題点を解決するため、次の手段を用いる。

すなわち、３２ビットマイクロコンピュータと

メモリからなる暗号変換装置において、

暗号変換されるべきデータの換字変換処理を、３２ビットのデータｘとｙ同士の演算、

$$x + y,$$

つまり、ｘとｙを加算し、2^{32}で割った余りをとるという処理と、

$$Rot_2(x) + x + \alpha,$$

つまり、３２ビットのデータｘを左または右に２ビット循環シフトした後、その結果得られたデータとｘと一定値αを加算した後、2^{32}で割った余りをとるという処理

を組み合わせることにより実現し、

暗号変換されるデータの転置変換処理を、

$$Rot_4(x),$$

すなわち、３２ビットのデータｘを左または右へ４ビット循環シフトするという処理と、

$$Rot_8(x),$$

すなわち、３２ビットのデータｘを左または右ｘへ８ビット循環シフトするという処理と、

$$Rot_{16}(x),$$

すなわち、３２ビットのデータｘを左または右へ１６ビット循環シフトするという処理

を組み合わせることにより実現する。

〔作用〕

これにより、３２ビットマイクロコンピュータを用いて、１回の基本命令で３２ビットのデータが換字または転置されるので、暗号変換を高速に行うことができる。

〔実施例〕

第１図は、本発明の一実施例である。

第１図において、６４ビットの平文１０１と６４ビット×４＝２５６ビットの鍵データ１００が３２ビットマイクロコンピュータに入力され、その後、プログラム１０３内の命令の順に３２ビットマイクロコンピュータ１０２において暗号変換され、その結果として６４ビットの暗号文１０４が出力される。

第２図は、第１図の３２ビットマイクロコンピュータ１０２とプログラム１０３において実行される暗号変換処理のフローを示している。

２０１：入力されたデータＭは上位３２ビットM_1と下位３２ビットM_2に分割される。

２０２：M_1とM_2のビット対応の排他的論和がとられる。

$$WORK2 \leftarrow M_1 \oplus M_2$$

以下、＋は同様の処理を示すものとする。

２０３：ＷＯＲＫ２と鍵データK_1のモジュロ加算が行われる。

$$x \leftarrow WORK2 + K_1$$

ここに、$x + K_1$はxとK_1の和を2^{32}で割った余りをとるという、2^{32}を法としたモジュロ加算を示している。

以下、＋は同様の処理を示すものとする。

２０４：ｘを左へ２ビット循環シフトした後、そのデータとｘと１のモジュロ加算をとる。

$$x \leftarrow Rot_2(x) + x + 1$$

以下、Rot_2は同様の処理を示すものとする。

１０５：ｘを左へ４ビット循環シフトした後、そのデータとｘとの排他的論理和をとる。

$$x \leftarrow Rot_4(x) \oplus x$$

以下、Rot_4は同様の処理を示すものとする。

２０６：$WORK1 \leftarrow x \oplus M_1$

２０７：$x \leftarrow x + K_2$

２０８：$x \leftarrow Rot_2(x) + x + 1$

$y \leftarrow x$

２０９：$x \leftarrow Rot_8(x) \oplus x$

ここに、$Rot_8(x)$ はｘを左へ８ビット循環シフトさせることを示す。

２１０：$x \leftarrow x + K_3$

２１１：$x \leftarrow Rot_2(x) + x + 1$

２１２：$x \leftarrow Rot_{16}(x) + (x \wedge y)$

ここに、$Rot_{16}(x)$ はｘを左へ１６ビット循環シフトすることを示す。また、$x \wedge y$ はｘとｙとのビット対応の論理積をとることを示す。

２１３：ＷＯＲＫ２←ｘ⊕ＷＯＲＫ２

２１４：$x \leftarrow WORK2 + K_4$

２１５：$x \leftarrow Rot_2(x) + x$

２１６：ＷＯＲＫ１←ＷＯＲＫ１⊕ｘ

２１７：ＷＯＲＫ２←ＷＯＲＫ２⊕ＷＯＲＫ１

２１８：ＷＯＲＫ１を出力データの上位３２ビット、ＷＯＲＫ２を出力データの下位３２ビットとして出力する。

以上、第２図に示すように関数π_1～π_4を定義すると、本実施例は、

$$C = \pi_1 \cdot \pi_2 \cdot \pi_3 \cdot \pi_4 \cdot \pi_1 \cdot \pi_4 \cdot \pi_3 \cdot \pi_2 \cdot \pi_1 (M)$$

としてもよい。

このとき、復号変換の式は

$$M = \pi_1 \cdot \pi_2 \cdot \pi_3 \cdot \pi_4 \cdot \pi_1 \pi_1 \cdot \pi_2 \cdot \pi_3 \cdot \pi_4 \cdot \pi_1 (M)$$

である。

同様に、一般に本実施例をｎ回繰り返したものを暗号変換としてもよい。

実施例の変形例２

第４図は、本発明の他の実施例である。

４０１：入力されたデータＭは上位１６ビットM_1と下位１６ビットM_2に分割される。

４０２：M_1とM_2のビット対応の排他的論理和がとられる。

$$C = \pi_1 \cdot \pi_4 \cdot \pi_3 \cdot \pi_2 \cdot \pi_1 (M)$$

というように合成関数で表すことができる。

関数$\pi i \cdot \pi i$（ｉ＝１～４）は、すべて、

$$\pi i \cdot \pi i (x) = x$$

というように同じ関数変換を２回繰り返すともとに戻るていう性質がある。

したがって、復号関数として、

$$M = \pi_1 \cdot \pi_2 \cdot \pi_3 \cdot \pi_4 \cdot \pi_1 (C)$$

を用いれば、暗号文Ｃをもとの平文Ｍに戻すことができる。

実施例の変形例１

本実施例を２回繰り返したものを暗号変換として用いてもよい。すなわち、暗号変換を、

$$WORK2 \leftarrow M_1 \oplus M_2$$

以下、＋は同様の処理を示すものとする。

４０３：ｘと鍵データK_1のモジュロ減算が行われる。

$$x \leftarrow x - K_1$$

ここに、$x - K_1$はｘとK_1の差を2^{16}で割った余りをとるという、2^{16}を法としたモジュロ減算を示している。

以下、－は同様の処理を示すものとする。

４０４：ｘを左へ２ビット循環シフトした後、そのデータとと１のモジュロ減算を行う。

$$x \leftarrow Rot(x) - x - 1$$

以下、Rot_2は同様の処理を示すものとする。

４０５：ｘを左へ４ビット循環シフトした後、そのデータとｘとの排他的論理和をとる。

$$x \leftarrow Rot_4(x) \oplus x$$

以下、Rot_4は同様の処理を示すものとする。

406：WORK1 ← x ⊕ M_1

407：x ← x ← K_2

y ← x

408：x ← Rot_2(x) − X − 1

409：x ← Rot_8(x) − (x ∧ y)

ここに、Rot_8(x)はxを左へ8ビット循環シフトすることを示す。また、x ∧ yはxとyとのビット対応の論理積をとることを示す。

410：WORK2 ← x ⊕ WORK2

411：x ← WORK2 − K_3

412：x ← Rot_2(x) − x − 1

413：WORK1 ← WORK1 ⊕ x

414：WORK2 ← WORK2 ⊕ WORK1

415：WORK1を出力データの上位16ビット、WORK2を出力データの下位16ビットとして出力する。

504：xを左へ2ビット循環シフトした後、そのデータとxと1のモジュロ加算を行う。

$$x \leftarrow Rot_2(x) + x + 1$$

以下、Rot_2は同様の処理を示すものとする。

505：x ← Rot_4(x) + (x ∧ y)

ここに、Rot_4(x)はxを左へ4ビット循環シフトすることを示す。また、x ∧ yはxとのビット対応の論理積をとることを示す。

506：WORK1 ← WORK1 ⊕ x

507：x ← WORK1 + K_2

508：x ← Rot_4(x) + x + 1

509：WORK2 ← WORK2 ⊕ x

510：WORK1 ← WORK1 ⊕ WORK2

511：WORK1を出力データの上位8ビット、WORK2を出力データの下位8ビットとして出力する。

実施例の変形例4

第6図は本発明の他の一実施例である。

実施例の変形例3

第5図は、本発明の他の実施例である。

501：入力されたデータMは上位8ビットM_1と下位8ビットM_2に分割される。

502：M_1とM_2のビット対応の排他的論和がとられる。

$$WORK2 \leftarrow M_1 \oplus M_2$$

以下、＋は同様の処理を示すものとする。

503：xと鍵データK_1のモジュロ加算が行われる。

$$x \leftarrow WORK2 + K_1$$

$$y \leftarrow x$$

ここに、x + K_1はxとK_1の差を2^8で割った余りをとるという、2^8を法としたモジュロ加算を示している。

以下、＋は同様の処理を示すものとする。

（1）認証を行うメッセージ62を鍵データとして、任意の初期値61を本発明によるアルゴリズム63を用いて暗号化する。

（2）暗号結果64を、（1）において用いたメッセージの続きのデータにより再び暗号化し、メッセージの終わりまでこの操作を繰り返す。

（3）最後的な暗号結果をメッセージ認証コード65として出力する。

実施例の変形例5

第7図は本発明の他の実施例である。本ICカードは、第7項記載の方式によりメッセージの認証コードを生成する。

（1）メッセージの認証を行うために必要な初期値76をI／O74を通して、ICカード71内のマイクロコンピュータ72に送信する。

（2）認証を行うメッセージ77を（1）と同様にマイクロコンピュータ72に順次送信し、マイクロコンピュータ72は、メモリ73に記憶されている暗号ソフト75により認証コード78を生成する。

〔効果〕

本実施例は、第３図に示すような換字、転置の繰返しを行っている。

つまり、第２図に示す実施例、（２０３、２０４）、（２０７、２０８）、（２１０、２１１）、（２１４、２１５）の処理は、

$$x \leftarrow x + Ki$$
$$x \leftarrow Rot_2(x) + (x) + 1$$

の形となっており、これは、それぞれ、３２ビットのデータを４ビットずつのブロックに分割したとき、各ブロック単位の換字処理を、上記２回のデータ変換により８ブロック分一斉に行っていると見ることができる。

ここに、４ビットのブロックデータ

$A = (a_1, a_2, a_3, a_4)$、ただし、

$a_i = 1$ or 0 $(i = 1 \sim 4)$

が、

$B = (b_1, b_2, b_3, b_4)$、ただし、

$b_i = 1$ or 0 $(i = 1 \sim 4)$

に換字変換されるということは、

ブール代数の演算f_1、f_2、f_3、f_4が存在して、

$$b_1 = f_1(a_1, a_2, a_3, a_4)$$
$$b_2 = f_2(a_1, a_2, a_3, a_4)$$
$$b_3 = f_3(a_1, a_2, a_3, a_4)$$
$$b_4 = f_4(a_1, a_2, a_3, a_4)$$

となることを示す。

また、第２図の２０５、２０９、２１２はそれぞれ、

（１）$x \leftarrow Rot_4(x) \oplus x$

（２）$x \leftarrow Rot_8(x) \oplus x$

（３）$x \leftarrow Rot_{16}(x) + (x \wedge y)$

の処理を行っており、これらは、それぞれ、

（１）４ビット左循環シフトを行うという転置を行った後、さらに換字を行うという処理、

（２）８ビット左循環シフトを行うという転置を行った後、さらに換字を行うとう処理、

（２）１６ビット左循環シフトを行うという処理を示している。

第３図から明らかなように、最初の３２ビットのデータのうち、いかなるビットの変化も最後の３２ビットのデータすべてに影響を与えることが分かる。

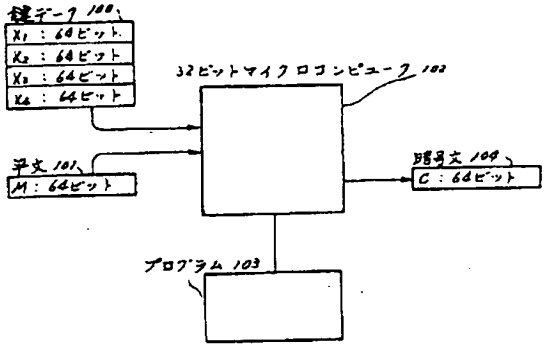
これにより、本実施例は、高度なランダム性を持つ暗号変換を効率良く行うという効果が得られることが分かる。

４．図面の簡単な説明

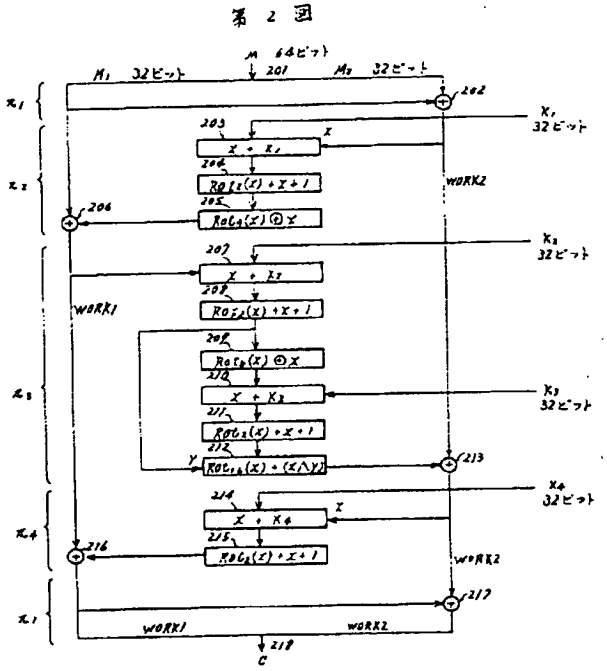
第１図は、本発明を実施する暗号変換装置の一実施例、第２図は、第１図における暗号変換の詳細を示すフローチャート、第３図は、本発明の実施例が効率的に換字変換、転置変換を繰り返していることを示す説明図、第４図は、１６ビットマイクロコンピュータを用いた場合の暗号変換の詳細を示すフローチャート、第５図は、８ビットマイクロコンピュータを用いた場合の暗号変換の詳細を示すフローチャート、第６図は、本発明による暗号アルゴリズムを用いてメッセージ認証コードを生成する方法を示すフローチャート、第７図は、本発明による暗号アルゴリズムを用いてメッセージ認証コードを生成するＩＣカードの構成図である。

代理人弁理士　小　川　勝　男

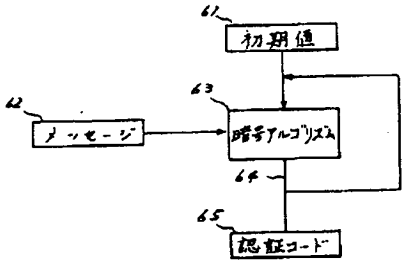
第 1 図



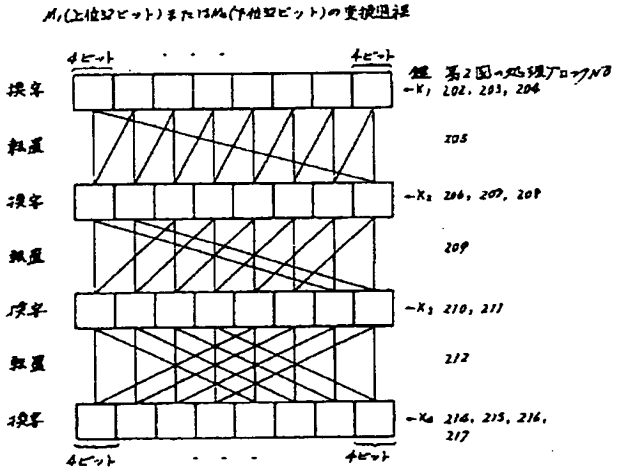
第 2 図



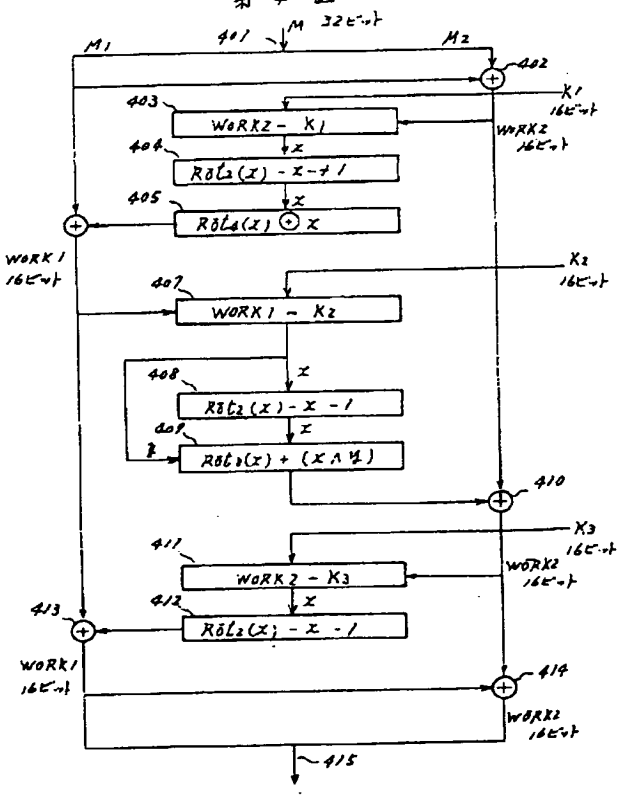
第 6 図



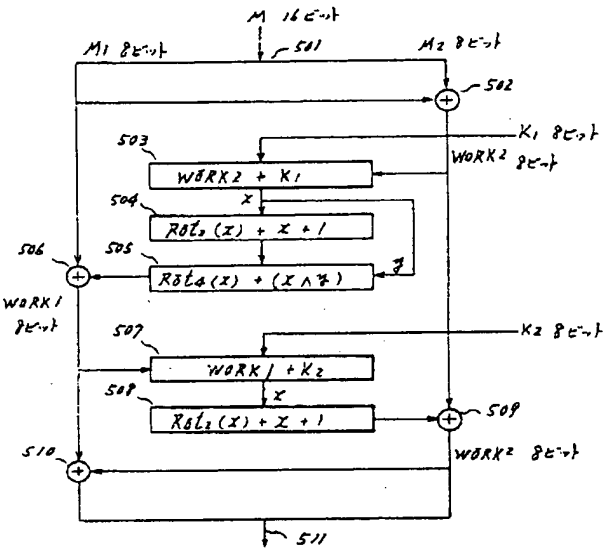
第 3 図



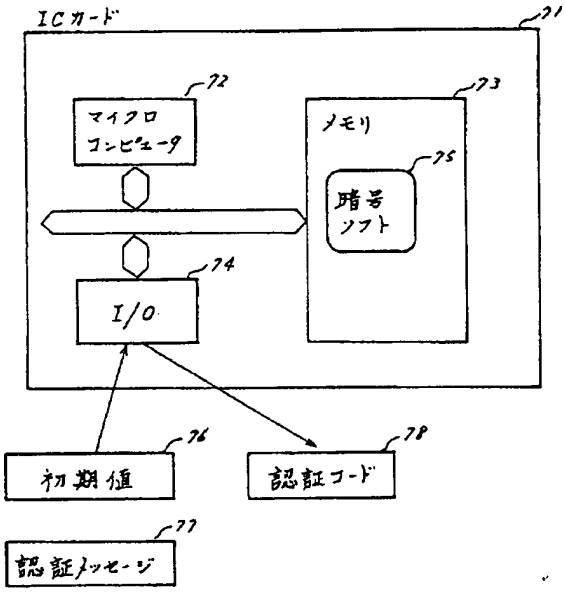
第 4 図



第 5 図


M 16ビット
501
M1 8ビット
M2 8ビット
502
K1 8ビット
503
WORK2 + K1
WORK2 8ビット
X
504
ROL3 (X) + X + 1
505
ROL4 (X) + (X∧Y)
Y
506
WORK1 8ビット
K2 8ビット
507
WORK1 + K2
X
508
ROL2 (X) + X + 1
509
WORK2 8ビット
510
511

第 7 図


ICカード
71
72
マイクロコンピュータ
73
メモリ
75
暗号ソフト
74
I/O
76
初期値
78
認証コード
77
認証メッセージ

【公報種別】特許法第１７条の２の規定による補正の掲載
【部門区分】第６部門第２区分
【発行日】平成８年（１９９６）７月１２日

【公開番号】特開平１－２７６１８９
【公開日】平成１年（１９８９）１１月６日
【年通号数】公開特許公報１－２７６２
【出願番号】特願昭６３－１０３９１９
【国際特許分類第６版】
G09C 1/00 9364-5L

手続補正書

平成 7年 4月28日

特許庁長官 殿

事件の表示

昭和６３年 特許願 第１０３９１９号

発明の名称 暗号化方法及び暗号の復号化方法

補正をする者
事件との関係 特許出願人
名称 (510) 株式会社 日立製作所
（ほか１名）

代理人
住所 〒100 東京都千代田区丸の内一丁目５番１号
株式会社 日立製作所内
電話 東京 3212－1111(大代表)
氏名 (6850) 弁理士 小川勝男

補正の対象 明細書の「発明の名称」の欄、「特許請求の範囲」の欄、及び「発明の詳細な説明」の欄。

補正の内容
1．明細書の発明の名称の欄の記載を「暗号化方法及び暗号の復号化方法」に訂正する。
2．明細書の特許請求の範囲の欄の記載を別紙のとおりに補正する。
3．明細書の発明の詳細な説明の欄について以下の補正を行う。
（1）明細書第１６頁第１３行「本実施例を」を「上記実施例における変換関数 π_1 から π_4 までにあたる処理を」に訂正する。
（2）明細書第１７頁第１～２行の式を次のとおりに訂正する。
「 $C = \pi_1 \cdot \pi_4 \cdot \pi_3 \cdot \pi_2 \cdot \pi_1 \cdot \pi_4 \cdot \pi_3 \cdot \pi_2 \cdot \pi_1(M)$ 」
（3）同頁第５～６行の式を次のとおりに訂正する。
「 $M = \pi_1 \cdot \pi_2 \cdot \pi_3 \cdot \pi_4 \cdot \pi_2 \cdot \pi_3 \cdot \pi_2 \cdot \pi_4 \cdot \pi_2(C)$ 」

以上

別紙

特許請求の範囲

1．暗号化対象データに対して所定ビット長のブロックごとに暗号化処理を行う暗号化方法において、鍵データをそれぞれ用いてデータへ換字処理を施すこと、およびデータ撹乱のためデータに転置処理を施すことを交互に実施し、上記転置処理は、2^n ビット（$n=2$ 以上の整数）だけ右もしくは左に処理中のデータを循環シフトする操作を含むことを特徴とする暗号化方法。

2．上記転置処理における循環シフトの深さは換字処理と転置処理の交互実施ごとに 2^2 ビット、2^3 ビット、・・・、と順次倍増することを特徴とする特許請求の範囲第1項記載の暗号化方法。

3．上記転置処理は、32ビットのデータを4ビット、または8ビット、または16ビットだけ右もしくは左に循環シフトする操作を含むことを特徴とする特許請求の範囲第1項記載の暗号化方法。

4．上記換字処理と転置処理を組み合わせた暗号化処理は、暗号化されたデータと、次の暗号化されるべきデータ戸の合いだで演算処理を施した後、該演算処理結果をさらに暗号化するというフィードバック処理を有することを特徴とする特許請求の範囲第1項記載の暗号化方法。

5．鍵データをそれぞれ用いてデータへ換字処理を施すこと、およびデータ撹乱のためデータに転置処理を施すことを交互に実施し、上記転置処理は、2^n ビット（$n=2$ 以上の整数）だけ右もしくは左に処理中のデータを循環シフトする操作を含む所定の暗号化方法を用い、デジタル署名により認証すべきメッセージを順次切り出して上記鍵データとし、所定の初期値を上記所定の暗号化方法で順次暗号化してメッセージ承認コードとするメッセージの認証方法。

6．鍵データをそれぞれ用いてデータへ換字処理を施すこと、およびデータ撹乱のためデータに転置処理を施すことを交互に実施し、上記転置処理は、2^n ビット（$n=2$ 以上の整数）だけ右もしくは左に処理中のデータを循環シフトする操作を含む所定の暗号化アルゴリズムをＩＣカードの記憶領域に記憶し、上記ＩＣカードでは認証すべきメッセージと初期データとが入力すると、該メッセージを順次切り出して上記鍵データとし、上記初期データに対し上記所定の暗号化アルゴリズムによる暗号化処理を順次施してメッセージ承認コードを発生することを特徴とするメッセージの認証方法。

7．鍵データをそれぞれ用いてデータへ換字処理を施すこと、およびデータ撹乱のためデータに転置処理を施すことを交互に実施し、上記転置処理は、2^n ビット（$n=2$ 以上の整数）だけ右もしくは左に処理中のデータを循環シフトする操作を含む所定の暗号化方法で作成された暗号を復号する復号化方法であり、上記所定の暗号化方法の処理ステップ全体をそれぞれ同一の関数変換を2度行えばデータが元に戻る関数変換の複数段階の積み重ねとしたとき、上記作成された暗号に対し、上記複数段階の関数変換を上記暗号化方法とは逆の順序で実行することを特徴とする暗号の復号化方法。